```
from google.colab import files

uploaded = files.upload()
```

```
Chọn tệp Không có tệp nào được chọn  Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving c4-train.00000-of-01024-30K.json.gz to c4-train.00000-of-01024-30K.json.gz
Saving requirements.txt to requirements (1).txt
```

```
import gensim.downloader as api
model = api.load("glove-wiki-gigaword-50")
print("Model loaded:", model)
```

```
[==================================================] 100.0% 66.0/66.0MB downloaded
Model loaded: KeyedVectors<vector_size=50, 400000 keys>
```

```
pip install -r requirements.txt
```

```
Collecting flask==3.0.2 (from -r requirements.txt (line 1))
  Downloading flask-3.0.2-py3-none-any.whl.metadata (3.6 kB)
Collecting requests==2.31.0 (from -r requirements.txt (line 2))
  Downloading requests-2.31.0-py3-none-any.whl.metadata (4.6 kB)
Collecting pandas==2.2.1 (from -r requirements.txt (line 3))
  Downloading pandas-2.2.1-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (19 kB)
Requirement already satisfied: numpy==1.26.4 in /usr/local/lib/python3.12/dist-packages (from -r requirements.txt (line 4)) (1.26.4)
Requirement already satisfied: Werkzeug>=3.0.0 in /usr/local/lib/python3.12/dist-packages (from flask==3.0.2->-r requirements.txt (l
Requirement already satisfied: Jinja2>=3.1.2 in /usr/local/lib/python3.12/dist-packages (from flask==3.0.2->-r requirements.txt (line
Requirement already satisfied: itsdangerous>=2.1.2 in /usr/local/lib/python3.12/dist-packages (from flask==3.0.2->-r requirements.txt
Requirement already satisfied: click>=8.1.3 in /usr/local/lib/python3.12/dist-packages (from flask==3.0.2->-r requirements.txt (line
Requirement already satisfied: blinker>=1.6.2 in /usr/local/lib/python3.12/dist-packages (from flask==3.0.2->-r requirements.txt (li
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests==2.31.0->-r require
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests==2.31.0->-r requirements.txt (
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests==2.31.0->-r requirements
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests==2.31.0->-r requirements
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas==2.2.1->-r requirement
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas==2.2.1->-r requirements.txt (line
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas==2.2.1->-r requirements.txt (l
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from Jinja2>=3.1.2->flask==3.0.2->-r requ
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.8.2->pandas==2.2.1->-r re
Downloading flask-3.0.2-py3-none-any.whl (101 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 101.3/101.3 kB 3.8 MB/s eta 0:00:00
Downloading requests-2.31.0-py3-none-any.whl (62 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.6/62.6 kB 4.5 MB/s eta 0:00:00
Downloading pandas-2.2.1-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.7/12.7 MB 49.6 MB/s eta 0:00:00
Installing collected packages: requests, pandas, flask
  Attempting uninstall: requests
    Found existing installation: requests 2.32.4
    Uninstalling requests-2.32.4:
      Successfully uninstalled requests-2.32.4
  Attempting uninstall: pandas
    Found existing installation: pandas 2.2.2
    Uninstalling pandas-2.2.2:
      Successfully uninstalled pandas-2.2.2
  Attempting uninstall: flask
    Found existing installation: Flask 3.1.2
    Uninstalling Flask-3.1.2:
      Successfully uninstalled Flask-3.1.2
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the sou
google-colab 1.0.0 requires pandas==2.2.2, but you have pandas 2.2.1 which is incompatible.
google-colab 1.0.0 requires requests==2.32.4, but you have requests 2.31.0 which is incompatible.
datasets 4.0.0 requires requests>=2.32.2, but you have requests 2.31.0 which is incompatible.
google-adk 1.14.1 requires requests<3.0.0,>=2.32.4, but you have requests 2.31.0 which is incompatible.
tsfresh 0.21.1 requires scipy>=1.14.0; python_version >= "3.10", but you have scipy 1.13.1 which is incompatible.
Successfully installed flask-3.0.2 pandas-2.2.1 requests-2.31.0
```

```
import numpy as np

class WordEmbedder:
    def __init__(self, model_name: str = "glove-wiki-gigaword-50"):
        print(f"Loading model: {model_name} ...")
        self.model = api.load(model_name)
        print("Model loaded successfully!")

    def get_vector(self, word: str):
```

```
            if word in self.model:
                return self.model[word]
            else:
                print(f"'{word}' not in vocabulary (OOV).")
                return np.zeros(self.model.vector_size)

    def get_similarity(self, word1: str, word2: str):
        if word1 in self.model and word2 in self.model:
            return self.model.similarity(word1, word2)
        else:
            return None

    def get_most_similar(self, word: str, top_n: int = 10):
        if word in self.model:
            return self.model.most_similar(word, topn=top_n)
        else:
            return []
    def embed_document(self, document: str):
        tokens = document.lower().split()
        vectors = []
        for token in tokens:
            if token in self.model:
                vectors.append(self.model[token])

        if len(vectors) == 0:
            return np.zeros(self.model.vector_size)
        else:
            return np.mean(vectors, axis=0)
```

```
# test/lab4_test.py

embedder = WordEmbedder("glove-wiki-gigaword-50")

print("Vector for 'king':")
print(embedder.get_vector("king")[:10])  # in 10 phần tử đầu

print("\nSimilarity:")
print("king vs queen:", embedder.get_similarity("king", "queen"))
print("king vs man:", embedder.get_similarity("king", "man"))

print("\nMost similar to 'computer':")
print(embedder.get_most_similar("computer", top_n=10))

print("\nDocument embedding:")
doc_vec = embedder.embed_document("The queen rules the country.")
print(doc_vec[:10])
```

```
Loading model: glove-wiki-gigaword-50 ...
Model loaded successfully!
Vector for 'king':
[ 0.50451   0.68607  -0.59517  -0.022801  0.60046  -0.13498  -0.08813
  0.47377  -0.61798  -0.31012 ]

Similarity:
king vs queen: 0.7839043
king vs man: 0.53093773

Most similar to 'computer':
[('computers', 0.9165045022964478), ('software', 0.8814992904663086), ('technology', 0.852556049823761), ('electronic', 0.8125867843
```

```
Document embedding:
[ 0.06438001  0.43381    -0.779435    0.0075025   0.07915     0.20077899
 -0.2454325  -0.05369498 -0.00951262 -0.68774253]
```

Bonus Task

```
import os
import tarfile
from gensim.utils import simple_preprocess
from pathlib import Path
from gensim.models import Word2Vec


data_path = Path("UD_English-EWT.tar.gz")
```

```
def read_sentences():
    with tarfile.open(data_path, 'r:gz') as tar:
        for member in tar.getmembers():
            if member.isfile() and member.name.endswith(".conllu"):
                f = tar.extractfile(member)
                for line in f:
                    line = line.decode('utf-8')
                    tokens = simple_preprocess(line)
                    if tokens:
                        yield tokens

print("Training Word2Vec model...")
sentences = list(read_sentences())
model = Word2Vec(sentences=sentences, vector_size=100, window=5, min_count=2, workers=4)

os.makedirs("/content/results", exist_ok=True)
model.save("/content/results/word2vec_ewt.model")

print("Model saved!")

model_loaded = Word2Vec.load("/content/results/word2vec_ewt.model")
print("Model reloaded successfully!")
print(model_loaded.wv.most_similar("computer", topn=5))
```

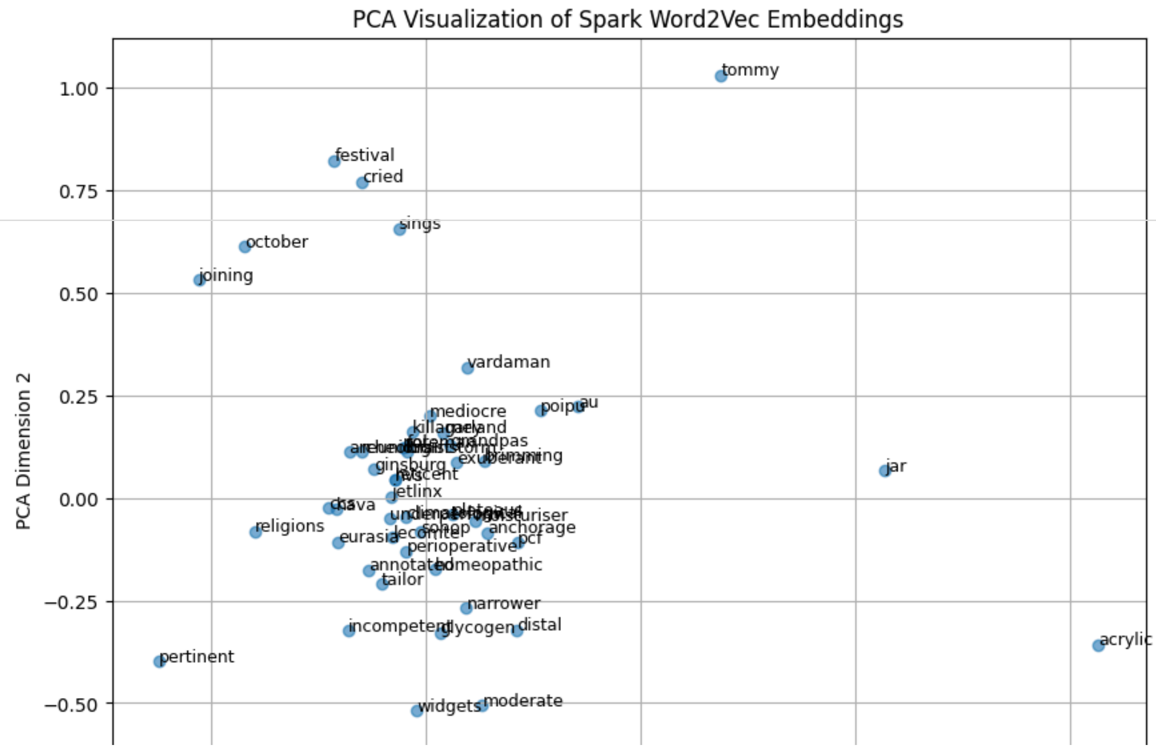
```
Training Word2Vec model...
Model saved!
Model reloaded successfully!
[('wheel', 0.815996527671814), ('campaign', 0.7777631878852844), ('cruise', 0.7764426469802856), ('budget', 0.7722913026809692), ('f
```

```
import gzip
import json
import matplotlib.pyplot as plt
from pyspark.sql import SparkSession
from pyspark.ml.feature import Word2Vec
from pyspark.sql.functions import lower, regexp_replace, split, col


spark = SparkSession.builder \
    .appName("SparkWord2VecDemo") \
    .config("spark.driver.memory", "4g") \
    .getOrCreate()

def read_json_gz(path):
    with gzip.open(path, 'rt', encoding='utf-8') as f:
        for line in f:
            try:
                obj = json.loads(line)
                if "text" in obj and obj["text"].strip():
                    yield obj["text"]
            except:
                continue

input_path = "c4-train.00000-of-01024-30K.json.gz"

rdd = spark.sparkContext.parallelize(read_json_gz(input_path))
df = rdd.map(lambda t: (t, )).toDF(["text"])

print(f"Loaded {df.count()} rows from {input_path}")

df = df.withColumn("text", lower(col("text")))
df = df.withColumn("text", regexp_replace(col("text"), "[^a-zA-Z\\s]", ""))
df = df.withColumn("words", split(col("text"), "\\s+"))
df = df.filter(col("words").getItem(0).isNotNull())

word2Vec = Word2Vec(vectorSize=100, minCount=5, inputCol="words", outputCol="result")
model = word2Vec.fit(df)
print("Word2Vec model trained successfully!")

try:
    synonyms = model.findSynonyms("computer", 5)
    print("\nMost similar words to 'computer':")
    synonyms.show(truncate=False)
except Exception as e:
    print(f"Could not find synonyms for 'computer': {e}")
```

```
model.write().overwrite().save("/content/spark_word2vec_model")
print("Model saved to /content/spark_word2vec_model")
```

```
Loaded 30000 rows from c4-train.00000-of-01024-30K.json.gz
Word2Vec model trained successfully!

Most similar words to 'computer':
+---------+------------------+
|word     |similarity        |
+---------+------------------+
|desktop  |0.705053985118866 |
|laptop   |0.6597256064414978|
|uwowned  |0.6360905766487122|
|computers|0.6352988481521606|
|usb      |0.6087394952774048|
+---------+------------------+

Model saved to /content/spark_word2vec_model
```

```
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
import pandas as pd

vocab = model.getVectors().sample(False, 0.01, seed=42).toPandas()
vectors = vocab['vector'].apply(lambda v: v.toArray()).tolist()

pca = PCA(n_components=2)
reduced = pca.fit_transform(vectors)

vocab['x'] = reduced[:, 0]
vocab['y'] = reduced[:, 1]

sample = vocab.sample(50, random_state=42)

plt.figure(figsize=(10, 8))
plt.scatter(sample['x'], sample['y'], alpha=0.6)
for i, word in enumerate(sample['word']):
    plt.annotate(word, (sample['x'].iloc[i], sample['y'].iloc[i]), fontsize=9)

plt.title("PCA Visualization of Spark Word2Vec Embeddings")
plt.xlabel("PCA Dimension 1")
plt.ylabel("PCA Dimension 2")
plt.grid(True)
plt.show()

spark.stop()
```


PCA Visualization of Spark Word2Vec Embeddings
PCA Dimension 2
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
tommy
festival
cried
sings
october
joining
vardaman
mediocre
poipu
au
jar
ginsburg
jetlinx
religions
eurasia
anchorage
perioperative
annotated
homeopathic
tailor
narrower
incompetent
glycogen
distal
acrylic
pertinent
widgets
moderate